**Web前端Vue3阶段测试考试试卷**

**总分100分 考试时间：120分钟 考试形式：闭卷**

**一、选择题（每题1分，共20分）**

1.在Vue实例中可以定义方法的是（ B ）选项。

A. el B. methods C. data D. computed

2.下列指令中，不能实现条件判断的是（ D ）。

A. v-if B. v-else

C. v-show D. v-bind

3.下列指令中，可以对表单元素进行双向数据绑定的是（ A ）。

A．v-model B．v-on

C．v-bind D．v-show

4.Vue子组件中触发自定义事件的方法是（ A ）。

A．this.$emit B．this.$event C．$this D．this.$on

5.在元素显示与隐藏的过渡中，用于定义元素离开过渡结束状态的class类名是（ C ）。

A．v-enter-from B．v-enter-active

C．v-leave-to D．v-leave-active

6.下列哪个不是webpack配置文件中的配置选项？（ D ）。

A．mode B．entry

C．module D．input

7.下列关于Vue生命周期钩子函数的说法错误的是（ D ）。

A．beforeCreate钩子在Vue实例创建之前调用

B．created钩子在Vue实例创建完成后进行调用

C．mounted钩子在DOM文档挂载完毕后调用

D．destroyed钩子在实例被销毁之前调用

8.Vue定义number=1; str=‘hykj’; boo=true,下面双大括号中，哪个是无效的表达式？（ D ）

A．{{number + 10}} B．{{str.toUpperCase()}}

C．{{boo ? '真' : '假'}} D．{{if(boo) return 1}}

9.Vue使用 v-for 指令不能遍历下列哪种数据类型？（B ）。

A．字符串 B．布尔 C．数组 D．对象

10.Vue如果要阻止浏览器的默认行为，需要使用（ B ）事件修饰符。

A..stop B.prevent C.self D.once

11.在Vue中，调用下面数组的方法，哪个不会触发视图的更新？（ D ）。

A.push B.unshift C.splice D.filter

12.下列关于Vue3组件的描述错误的是（C ）。

A. 局部注册的组件只能在当前组件中使用，无法在其它组件中使用

B．组件选项对象中data，methods,computed,watch中this都指向Vue实例

C．SFC组件的模板只能有一个根元素

D．注册的全局组件可以全局直接使用。

13.下列关于组件 Prop 的描述错误的是（C ）。

A. Prop 是父组件向子组件传递数据的，需在子组件定义Props属性接收 。

B．使用 Prop 传递数据除了可以是数值和字符串类型之外，还可以是数组或对象类型。

C．使用 Prop 验证的时候，Prop 接收的参数是一个数组 。

D．如果 Prop 传递的是一个对象或数组，那么它是按引用传递的。

14.下列关于路由的描述错误的是（ C ）。

A. 通过router-link组件的 to 属性可以设置目标地址来切换路由。

B．在组件的template中支持路由的嵌套。

C．使用 router 的 to()方法也可以实现导航的功能

D．如果为要访问的路径设置了重定向规则，则访问该路径时会被重定向到指定的路径

15.下列关于Options API的描述错误的是（ C ）。

A．在Vue3中data必须是一个函数, 函数会返回一个对象。

B．data选项中定义的数据最终会放到响应式系统中。

C．methods属性是一个函数，可以在该函数中定义多个方法。

D．watch选项可以用来侦听数据的变化，并支持deep深度侦听。

16.下列关于computed和methods选项描述错误的是（ C ）。

A．计算属性和方法都可以通过this来访问到data中定义的数据。

B．计算属性会基于它们的依赖关系进行缓存，而方法不会。

C．在方法中使用箭头函数定义，可以通过this来拿到Vue的实例。

D．计算属性中的依赖的数据发生变化，会重新进行计算。

17.下列关于setup函数描述错误的是（ D ）。

A．在Vue3中可以在setup函数的第一个参数中拿到组件的props属性。

B．在Vue3中可以在setup函数的第二个参数中拿到组件的context对象。

C．setup函数可定义响应式数据、方法和计算属性，并在该函数中return给模板用。

D．setup中可以通过this来拿到Vue的实例。

18.下列哪一项不是 Vue 的生命周期函数？（ B ）

A．beforeCreate B．methods C．beforeMount D．Mounted

19.下列哪一项不是 Vuex 的组成部分？（ B ）

A．getters B．setters C．mutations D．actions

20.下列关于Vuex的描述错误的是（ D ）。

A．state函数返回一个包含定义状态的对象，在该对象定义的数据是响应式的。

B．mutations 定义改变state的方法， 在组件中通过commit调用。

C．actions可定义同步和异步的函数，在组件中通过 dispatch方法调用。

D．getters 可定义一些对state数据操作的函数，这些函数相当于组件中的methods。

**二、填空题（每空1分，共20分）**

1.Vue中MVVM框架主要由哪3部分组成（1）。

Model View ViewModel

2.Vue的哪个指令可以将某个属性绑定到元素上？（2）。

v-bind

3.小程序如何给View组件绑定事件并给事件对象传递数据？（3）。

<view bindtap=”onViewClick” data-id=”{{100}}”></view>

4.Vue事件传递有冒泡和捕获(默认是冒泡)，那Vue中如何阻止事件的冒泡（4）。

<div @click.stop=”handleClick”>button</div>

5.在Vue3中获取组件的实例和元素的实例的API是（5）。

<div ref =“divRef” ></div>

6.在script setup语法中需用哪些API来定义props和emit（6）。

defineProps() defineEmits()

7.Vue中注册组件有哪几种？（7）。

全局注册 和 局部注册

8.Vue3中是使用哪个API来安装插件的？（8）。

app.use()

9.Vue3中响应式API有哪些（9）。

ref reactive toRefs computed watch watchEffect ....

10.小程序中响应式的单位是?（10）。

rpx

11.Vue中的那个内置组件可以将组件缓存起来？（11）。

keep-alive

12.Vue3组件可以使用哪个API来向部暴露属性和方法？（12）。

defineExpose()

13.安装Vue Router插件时，会注册哪两个全局组件？（13）。

router-link router-view

14.Vue3中可通过useRouter()拿到路由对象，该对象有哪些常用的方法？（14）。

router.go() router.back() router.push() router.addRouter() ...

15.请写出Vue Router中常用的路由守卫的API，至少2个？（15）。

beforeRouteLeave beforEach beforRouterUpdate afterEach ....

16.Pinia状态管理的Store有哪几个核心功能?（16）。

State Getters Actions

17.Vue项可以通过哪些脚手架来创建？（17）。

Vue-CLI create-app

18.如何解构reactive对象而不丢失响应？（18）。

toRefs()

1. Vue中给元素绑定click事件时如何获取event参数？（19）。

<div @click=”handleClick($event)”></div>

1. Vue Router4中支持哪几种路由模式？（20）。

history hash

**三、简答题（60分）**

**1.说出v-if 和 v-show 的区别? (5分)**

1）在用法上的区别：

v-show是不支持template；

v-show不可以和v-else一起使用；

2）本质的区别

v-show元素无论是否需要显示到浏览器上，它的DOM实际都是有存在的，只是通过CSS的display属性来进行切换；

v-if当条件为false时，其对应的原生压根不会被渲染到DOM中

1. **v-for循环中的key有什么作用？(5分)**

答：

1）有key的操作:

Vue框架在diff算法时，会根据key找到之前的VNode进行复用; 没找到有VNode可以复用, 再创建新的VNode

2）没有key的操作:

在diff算法, 如VNode一样就直接复用（不会根据key来寻找可复用节点），如不一样就进行移除或创建，从而导致VNode复用性不强了。

1. **Vue组件之间的通信有几种方案？至少写出4种。(5分)**

答：

1.父子通讯

props ：父组件传递给子组件

emit: 子组件传递给父组件

ref 组件对象：父组件拿到子组件对象，调用子组件暴露的属性和方法等

2）非父子通讯

A全局事件总线

先安装库 npm install hy-event-store

封装一个工具eventbus.js

在组件A中 通过eventBus.on监听事件

在组件B中通过eventBus.emit发射事件

B.Provide/Inject用于非父子组件之间共享数据

父组件有一个 provide 选项来提供数据

子组件有一个 inject 选项来开始使用这些数据(注入)

C.Vuex /Pinia 状态管理

1. **写出学过的Options API, 并说出每个选项的作用？至少写出5个。(5分)**

答：

1.data 定义数据

data在Vue3.x必须是一个函数, 函数会返回一个对象。

data返回的对象, 会被Vue进行劫持(放到响应式系统中), 所以data的数据发生改变时, 界面会重新渲染

2.methods 定义方法

methods属性是一个对象 -> 定义很多方法--->这些方法可以绑定到模板上

在该方法中，我们可以使用this关键字来直接访问到data中返回的对象的属性

里面函数不能是箭头函数，如果是箭头函数,因为箭头函数不绑定this,所以它会在上层作用域中查找this, 查找到this-->window

3.computed 计算属性

对于任何包含响应式数据的复杂逻辑，你都应该使用计算属性

可以通过this访问数据

计算属性会基于它们的依赖关系进行缓存

在数据不发生变化时，计算属性是不需要重新计算的

如果依赖的数据发生变化，在使用时，计算属性依然会重新进行计算

4.watch 侦听器

用来监听某个数据的变化

deep 深度监听

immediate 第一次渲染直接执行一次监听器

5.components

注册局部组件

该components选项对应的是一个对象，对象中的键值对是 组件的名称: 组件对象

6.props 完成父子组件之间通信

props是你可以在组件上注册一些自定义的attribute

父组件给这些attribute赋值，子组件通过attribute的名称获取到对应的值

7.emits

记录发射了哪些自定义事件

emits 可以是数组或对象，从组件触发自定义事件，emits 可以是简单的数组，也可以是对象，后者允许配置事件验证

8.provide 用于非父子组件之间共享数据

父组件有一个 provide 选项来提供数据

9.inject 用于非父子组件之间共享数据

子组件有一个 inject 选项来开始使用提供provide的数据

10.常见生命周期函数

beforeCreate :组件实例在创建之前

created: 组件被创建完成

可以发送网络请求

可以事件监听this.$watch()

beforeMount : 组件template准备被挂载

mounted :组件template已经被挂载，可以获取DOM,可以使用DOM

1. **写出Vue3 中Composition API, 并说出每个API的作用，至少写5个。 (5分)**

答：

1. ref

包裹任意类型的值，将包裹的值加入响应式

2. reactive

包裹复杂类型的值，将包裹的值加入响应式

3. computed

把一些复杂逻辑用computed进行包裹，如同Options API中的计算属性一样

computed会自动收集相关依赖，当依赖发生变化时，会自动进行更新

4. watch

watch可以监听单个数据源，也可以监听多个数据源

watch是懒执行，第一次是不会执行的，除非你为其提供第三个参数中的immediate属性为true

watch只有等到监听的数据源发生了变化后，才会执行第二个参数（回调）

watch可以获取监听数据源的前后变化的值

侦听多个数据源的时候，第一个参数是数组类型

1. watchEffect

watchEffect会自动收集依赖，收集的依赖是第一个参数，也就是回调函数中有哪些东西是加入响应式的

如果这个值加入了响应式就会被收集起来，当被收集的值发生了变化，就会重新执行这个回调函数

watchEffect第一次执行是在DOM挂载前执行的，所以如果你想在第一次执行时拿到DOM元素

需要传入第二个参数，第二个参数是一个对象，让其flush属性的值为post即可

6. toRefs

对reactive进行解构后就失去了响应式的效果，因为reactive返回的是一个Proxy对象

对Proxy对象进行解构，拿到的是纯净的值，所以没有了响应式的效果

如果想要对reactive进行解构，需要对其包裹一个toRefs

这么做相当于为reactive中的每一个值包裹了一个ref

1. **什么是Vuex？Vuex包含哪些核心模块？(5分)**

答：

1）什么是Vuex：

Vuex 是一个专为 Vue.js 应用程序开发的状态管理模式。每一个 Vuex 应用的核心就是 store（仓库）。“store” 基本上就是一个容器，它包含着你的应用中大部分的状态 ( state )。

Vuex 的状态存储是响应式的。当 Vue 组件从 store 中读取状态的时候，若 store 中的状态发生变化，那么相应的组件也会相应地得到高效更新。

改变 store 中的状态的唯一途径就是显式地提交 (commit) mutation。这样使得我们可以方便地跟踪每一个状态的变化。

2）Vuex包括以下几个核心模块：

State：定义了应用状态的数据结构，可以在这里设置默认的初始状态。

Getter：允许组件从 Store 中获取数据，mapGetters 辅助函数仅仅是将 store 中的 getter 映射到计算属性。

Mutation：是唯一更改 store 中状态的方法，且必须是同步函数。

Action：用于提交 mutation，而不是直接变更状态，可以包含任意异步操作。

Module：允许将单一的 Store 拆分为多个 store 且同时保存在单一的状态树中。

1. **Vue3如何获取router和route对象，它们有什么区别？(10分)**

答：

1）获取router和route对象

import { useRouer, useRoute } from ‘vue-router’

const router = useRouter()

const route = useRoute()

2）它们之间的区别

route是路由信息对象，在Vue3中通过 useRoute 来获取。包括了path， params，hash，query，fullPath，matched，name等路由信息参数。

router是路由实例对象，Vue3中通过 useRouter 来获取。包括了路由跳转 方法、钩子函数等，比如：push、go、back、addRouter、beforeEnter等。

1. **Vue3增加了哪些新特性？(10分)**

答：

1.更现代化的源码管理方式

Vue3对项目的管理进行了重大的重构，采用了monorepo的方式来进行管理：

2.代码使用TypeScript进行了重构

在Vue2.x的时候，它是使用Flow来进行类型的检测，但是Flow有很多的复杂场景是类型的支持并不是非常友好。所以在Vue3.x开始，整个项目全部使用了TypeScript进行重构，并且Vue3.x本身对TypeScript的支持也变得更加友好了。

3.使用Proxy进行数据劫持

在Vue2.x的时候，Vue2是使用Object.defineProperty来劫持数据的getter和setter方法。这种方式一直存在一个缺陷就是当给对象添加或者删除属性时，是无法劫持和监听。所以在Vue2.x的时候，不得不提供一些特殊的API，比如 $set 或 $delete，事实上这些都是一些hack方法，也增加了开发者学习新的API的成本。而在Vue3.x开始，Vue3使用Proxy来实现数据的劫持，这个API的用法和相关的原理会在后续讲到，后续会专门有文章讲解Vue3响应式的原理以及Proxy的用法。

4.编译阶段的优化

Vue3在编译阶段进行了很多的优化。生成Block Tree：在编译阶段对静态模板进行分析，生成Block Tree（主要是用来做性能优化）。 slot编译优化：Vue3优化了slot的生成，对于非动态slot中属性的更新只会触发子组件的更新。diff算法优化：Vue3相对于vue2在diff算法上也进行了优化。

5.Composition API

由Options API 到 Composition API。在Vue2.x的时候，我们会通过Options API来描述组件对象。Options API包括data、props、methods、computed、生命周期等等这些选项。Options API存在比较大的问题是多个逻辑可能是在不同的地方。比如created中会使用某一个method来修改data的数据，代码的内聚性非常差。Composition API可以将相关联的代码放到同一处进行处理，而不需要在多个Options之间寻找。

6.移除一些非必要的API

Vue3移除了实例上的$on， $off和 $once。也移除了一些特性：如filter、内联模板等。当然后面会讲对应的替代的方案。

1. **小程序有哪些传递数据的方法？(10分)**

答：

1）使用全局变量

在 app.js 中的 this.globalData = { } 中放入要存储的数据。

在组件.js 中， 头部 引入 const app = getApp(); 获取到全局变量app。

接使用 app.globalData.key 来进行赋值和获取值。

2）使用路由

wx.navigateTo 和 wx.redirectTo 时，可以通过在 url 后 拼接变量。

然后在 目标页面 通过在 onLoad 周期中，通过参数来获取传递过来的值。

3）使用本地缓存Storage

页面和组件之间数据的传递

使用第三方的事件总线库

5）使用 Mobx来管理小程序跨页面传递数据。